

Exploring Neural Ordinary Differential Equations as Interpretable Healthcare classifiers

Shi Li

Department of Computer Science
Columbia University
shili081100@columbia.edu

Abstract

Deep Learning has emerged as one of the most significant innovations in machine learning. However, a notable limitation of this field lies in the “black box” decision-making processes, which have led to skepticism within groups like healthcare and scientific communities regarding its applicability. In response, this study introduces a interpretable approach using Neural Ordinary Differential Equations (NODEs), a category of neural network models that exploit the dynamics of differential equations for representation learning. Leveraging their foundation in differential equations, we illustrate the capability of these models to continuously process textual data, marking the first such model of its kind, and thereby proposing a promising direction for future research in this domain. The primary objective of this research is to propose a novel architecture for groups like healthcare that require the predictive capabilities of deep learning while emphasizing the importance of model transparency demonstrated in NODEs.

1 Introduction

Deep learning has been recognized as a major innovation in machine learning (LeCun et al., 2015), transforming the analysis of data and the derivation of insights across various domains, including computer vision (Voulodimos et al., 2018) and natural language processing (Deng & Liu, 2018; Ono & Lee, 2024). Its capacity to learn hierarchical data representations renders it an influential tool, facilitating the automation of tasks previously deemed too complex or unattainable. Nonetheless, despite its significant contributions, deep learning presents notable challenges (Thompson et al., 2020). Among these is the issue of interpretability (Zhang et al. (2021), Chakraborty et al. (2017), Zhang & Zhu (2018)); particularly, models with intricate architectures tend to function as “black boxes”. The absence of transparency raises substantial concerns in critical areas such as healthcare ((Miotti et al., 2018), (Razzak et al., 2018)) and finance (Heaton et al., 2016), where the clarity of operations is imperative for ensuring trust, meeting ethical standards, and adhering to regulatory requirements (Wu et al., 2021).

In contrast with recent emerging NLP techniques (e.g., GPT, BERT, etc.), our focus centers on a specific category of neural networks referred to as Neural Ordinary Differential Equations (NODEs) (Chen et al., 2018), renowned for their exceptional performance across a multitude of data formats. Here, we present an extensive exploration of their utility within the textual modality, an area that has not been extensively explored. This paper advocates for the efficacy of NODEs in handling textual data, particularly in fields where transparency and model interpretability are crucial.

2 Related Works

2.1 Neural ODEs

Neural Ordinary Differential Equations (NODEs), introduced in 2018 by (Chen et al., 2018), have garnered widespread interest across various disciplines. These models offer a distinct advantage by continuously updating their internal state, facilitating seamless integration with time-series data and complex dynamic systems (Kidger, 2022). Demonstrating exceptional performance, NODEs have

Model	Interpretable?	Accuracy	F1	AUROC
Logistic Regression	yes	0.913	0.914	0.963
LGBM	yes	0.928	0.929	0.980
LSTM	no	0.932	0.933	0.929
BERT	no	0.940	0.943	0.948
Neural ODE	yes	0.930	0.932	0.937

Table 1: Benchmark of text classification.

found applications in tasks such as time-series forecasting (Kidger et al. (2020), Jin et al. (2022)), dynamic system modeling (Linot et al. (2023), Alvarez et al. (2020)), and function approximation (Ruiz-Balet & Zuazua, 2023). Furthermore, their interpretable nature and capability to capture continuous dynamics make them well-suited for healthcare applications, including medical image analysis and text-based outcome prediction. Although there has been limited exploration of time series analysis within Neural ODEs and healthcare (Qian et al. (2021), Gwak et al. (2020)), their potential in these other domains remains promising.

2.2 Interpretability

Interpretable deep learning refers to the development of machine learning models that produce results in a transparent and understandable manner (Li et al. (2022), Chen et al. (2019)). Unlike traditional deep learning models, which often function as "black boxes" with opaque decision-making processes, interpretable deep learning methods aim to provide insights into how and why a model arrives at a particular outcome. By incorporating features such as interpretable attention mechanisms (Choi et al., 2016), explainable embeddings (Subramanian et al., 2018), and model-agnostic interpretability techniques, interpretable deep learning enhances trust and confidence in the predictions generated by these models.

3 Methods

3.1 A Primer on Neural ODEs

Neural Ordinary Differential Equations (NODEs) fundamentally transform machine learning by treating the learning of continuous dynamics as solving an ordinary differential equation (ODE) (Kidger, 2022). Central to NODEs is the concept that the derivative of the hidden state of a neural network with respect to time, denoted as $\frac{d\mathbf{h}(t)}{dt}$, is a function f parameterized by a neural network. Formally, this relationship is expressed as $\frac{d\mathbf{h}(t)}{dt} = f(\mathbf{h}(t), t; \theta)$, where $\mathbf{h}(t)$ represents the hidden state at time t , and θ denotes the neural network parameters.

Given an initial condition $\mathbf{h}(t_0) = \mathbf{h}_0$, the solution to this ODE, $\mathbf{h}(t)$, at any time t is obtained by solving the integral $\mathbf{h}(t) = \mathbf{h}_0 + \int_{t_0}^t f(\mathbf{h}(s), s; \theta) ds$. This integral captures the evolution of the network's hidden state over time, allowing for modeling continuous-time dynamical systems without fixed-step discretization, as in traditional recurrent neural networks (RNNs).

To train a Neural ODE, the adjoint method (Zhuang et al., 2020) is commonly employed, a back-propagation technique that efficiently computes gradients without storing intermediate states. The adjoint state, denoted as $\mathbf{a}(t) = \frac{\partial L}{\partial \mathbf{h}(t)}$, where L is the loss function, evolves according to the ODE $-\frac{d\mathbf{a}(t)}{dt} = \mathbf{a}(t)^T \frac{\partial f(\mathbf{h}(t), t; \theta)}{\partial \mathbf{h}(t)}$, with the initial condition set at the end of the interval and integration performed backward in time. The gradient of the loss L with respect to the parameters θ is then obtained by integrating another term, $\frac{\partial L}{\partial \theta} = - \int_t^{t_0} \mathbf{a}(t)^T \frac{\partial f(\mathbf{h}(t), t; \theta)}{\partial \theta} dt$, over the same interval in the opposite direction. A basic example of a NODE implementation is provided in the appendix (Section A.1).

Admit Words	Home Words
old	patient
complaint	recieved
diagnostic	17
ambulance	walk-in
insurance	medications

Table 2: Top 5 words for each basing prediction

3.2 Current Gaps in Text Classification?

Deep learning-based models (RNNs, LSTMs, BERT, etc.) for text classification face significant challenges in terms of interpretability. These models are often perceived as black boxes due to their complex architectures and opaque decision-making processes. Understanding which features or words influence the model’s predictions is difficult, limiting the ability to trust and interpret the model’s outputs. Techniques such as attention mechanisms provide some insights into the model’s reasoning, but comprehensive explanations are often lacking. Furthermore, issues such as dataset biases and model errors can be challenging to diagnose and address without transparent interpretability tools.

3.3 How NODEs can model Text

In the context of our work, modeling text for classification using NODEs offers several advantages. Firstly, NODEs represent text as a continuous dynamic system, enabling a more natural representation compared to traditional approaches. They excel in capturing long-range dependencies in text, which can be challenging for RNNs. NODEs also provide flexibility in handling sequences of varying lengths, eliminating the need for padding or truncation. Moreover, their interpretable dynamics allow for a deeper understanding of how textual features evolve over time, crucial for tasks where interpretability is paramount. Additionally, NODEs can be trained efficiently using the adjoint method, making them scalable for training on large text corpora.

4 Results

4.1 Case Study: Hospital Outcome Prediction

Deep learning has garnered considerable attention in healthcare, spanning from predicting medical outcomes (Lee et al., 2024b; Jenkins et al., 2018; Lee et al., 2024a) to diagnosing diseases (Liu et al., 2018; Cheng et al., 2016), operationalization of medical tasks (Lee & Lindsey, 2024a) offering potential assistance across various clinical operations. However, lingering concerns about model interpretability persists (Stiglic et al., 2020; Wu et al., 2010). Transparency is particularly crucial, especially with the emergence of Generative AI techniques, given their propensity for producing hallucinations, which can be harmful in healthcare contexts (Lee & Lindsey, 2024b).

Therefore, in this case study, we employ NODEs for text classification, marking a pioneering application of this model type for such an objective. Framing this as a classification problem, our model adeptly learns the sequencing of text to discern the “dynamics” necessary for making predictions. For our study, we utilize classification objectives outlined in (Lee et al., 2024b) to predict Emergency Department admissions, with additional results on extended tasks provided in the appendix using text serialized from MIMIC-IV ED database (Johnson & et al., 2023). Before inputting the serialized text into the NODE, we preprocess it using TF-IDF.

To evaluate the effectiveness of the NODE, we compare its performance against two interpretable models, namely Logistic Regression and Light Gradient Boosted Machines (LGBM), as well as two non-interpretable models tailored for text processing, namely Long Short-Term Memory (LSTMs) and BERT. Our assessment focuses on the task of outcome prediction, and the results are presented in Table 4. This approach sheds light on the potential of NODEs in enhancing text classification tasks and underscores the importance of evaluating model interpretability alongside predictive performance.

Model	Interpretable?	Balanced Accuracy
SVM	somewhat	0.611
CNN	somewhat	0.781
VGG	no	0.729
Neural ODE	yes	0.689

Table 3: Benchmark of image classification.

4.1.1 Feature Importance with Saliency Maps

One advantage of the Neural ODE model over other attention-based approaches, such as BERT and LSTMs, is its ability to provide interpretable feature importance which can be directly mapped to words within the TF-IDF matrix via saliency maps. Saliency maps (Adebayo et al., 2018) offer an interpretable method to comprehend the decision-making process of machine learning models and reveal which words or sequences contributed most to the classification decision, thereby offering insights into the model’s reasoning. We therefore detail the saliency map from our hospital admission problem (Table 2), where specific keywords impacted the overall decision assigned by the model. This interpretability stands in contrast to the complex, hidden layers of BERT and recent LLMs, where understanding the precise influence of each token element on the output requires intricate analysis.

4.1.2 Vector Fields explain Decision Making

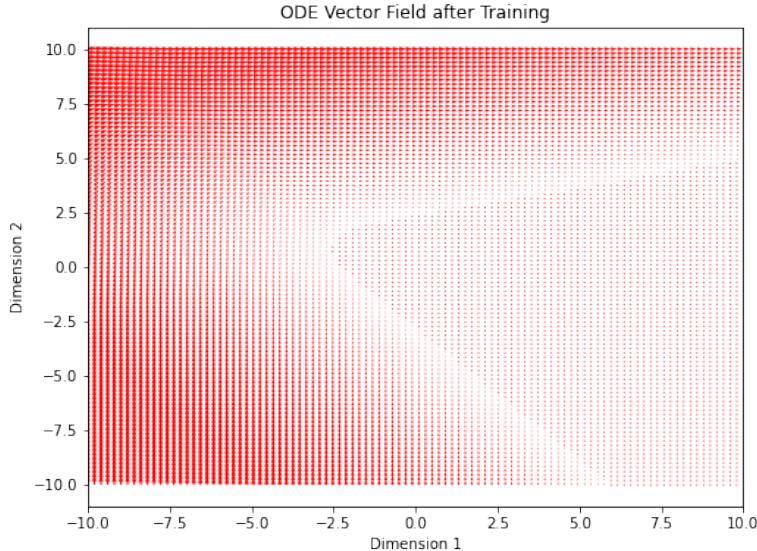


Figure 1: Visualizing the ODE Vector fields for a simple binary classification

Another method of visualization utilizes ODE vector fields within the NODEs architecture, enhancing the models’ interpretability (Figure 1). This approach enables intuitive visual demonstrations of high-dimensional data classification on a 2D plane. By modeling the dynamics of the neural network as an ODE, NODEs provide a vivid visualization of the hidden state’s evolution over time in reaction to input data. This evolution, captured in the vector field, visually represents the direction and magnitude of change at various points in the input space. By analyzing the vector field’s flow, insights into the model’s processing and classification of input data at attractors can be gained. Visualizing these trajectories allows for an intuitive understanding of how the model differentiates between data classes and makes classification decisions based on the NODE’s encoded dynamics.

4.2 Case Study: Predicting the Stage of Alzheimer’s

Another potential application of Neural Ordinary Differential Equations (ODEs) in healthcare is in medical imaging, where they can adopt a similar approach to the text classification example discussed earlier. Instead of analyzing sequences of text, Neural ODEs can be applied to sequences of pixels representing medical images. Image classification with Neural ODEs involves treating the pixels of an image as a sequence, similar to words in a sentence, and learning the dynamics of these pixels to distinguish unique classes. Mathematically, let $X(t)$ represent the state of the pixel sequence at time t , and $f(X(t), t; \theta)$ denote the continuous dynamics function parameterized by θ . The model continuously updates its internal state according to the differential equation: $\frac{dX}{dt} = f(X(t), t; \theta)$ as it processes the sequence of pixels. This equation captures the complex spatial relationships within the image over time. By integrating the principles of differential equations into the learning process, Neural ODEs offer a unique advantage in image classification tasks by leveraging the continuous and interpretable nature of their dynamics to make predictions.

5 Discussion

5.1 Continuous Modeling is Competitive to Discrete Modeling

Our results reveals that the NODEs performs comparably in accuracy to alternative models. However, continuous modeling approaches present distinct advantages over discrete methodologies in text classification tasks. Specifically, continuous modeling adeptly captures the evolving dynamics of text, facilitating the extraction of expressive and flexible features. This capability is particularly beneficial in contexts where understanding the nuanced aspects of language is essential for precise classification. Furthermore, unlike traditional neural networks, NODEs are not susceptible to the vanishing or exploding gradient problems (Section A.3), potentially ensuring more stable and effective training outcomes. By harnessing the continuous-time framework of the ODE, NODEs are adept at both interpolating and extrapolating text representations beyond the confines of observed data, significantly improving their generalization to new inputs.

5.2 Shaving Accuracy for Interpretability

One observation we made from our analysis was that there exists a trade-off between interpretability and accuracy with the NODEs architecture. Consequently, NODEs might compromise a degree of accuracy compared to more intricate deep learning architectures in NLP that place a higher emphasis on performance over interpretability, as evidenced by our findings. Although LSTMs and BERT models may be favored for their high accuracy, the value of NODEs lies in their interpretability.

5.3 Performance versus Memory Tradeoff

Compared to traditional discrete models, NODEs offer an additonal different trade-off in terms of time complexity and memory requirements. While the time complexity of training NODEs scales with the number of data points, the ODE solver steps, and the dimension of the state, it can be more efficient than the sequence-level computations required by full attention mechanisms, especially for long input sequences. Additionally, NODEs have the potential to be more memory-efficient, as they do not need to store the entire input sequence during the forward pass, but rather integrate the dynamics over time. This memory advantage can be particularly beneficial when working with large text corpora or resource-constrained environments. We see this to be quite the advantage as memory intensive models will lead to a more exclusive research community. We detail some of the mathematics regarding the convergence and time complexity further in the appendix (Section A.4, A.5)

6 Conclusion

This work has emphasized the potential of Neural ODEs in processing textual data, particularly noting their ability to address the challenges of transparency in deep learning architectures. We hope future researchers will use this work to build more sophisticated architectures for interpretable text based

classifiers even with the emergence of Large Language Model technologies since we only showcased this method at its simplest form.

References

- Julius Adebayo, Justin Gilmer, Michael Muelly, Ian Goodfellow, Moritz Hardt, and Been Kim. Sanity checks for saliency maps. *Advances in neural information processing systems*, 31, 2018.
- Victor M Martinez Alvarez, Rareş Roşca, and Cristian G Fălcuţescu. Dynode: Neural ordinary differential equations for dynamics modeling in continuous control. *arXiv preprint arXiv:2009.04278*, 2020.
- Supriyo Chakraborty, Richard Tomsett, Ramya Raghavendra, Daniel Harborne, Moustafa Alzantot, Federico Cerutti, Mani Srivastava, Alun Preece, Simon Julier, Raghuveer M Rao, et al. Interpretability of deep learning models: A survey of results. In *2017 IEEE smartworld, ubiquitous intelligence & computing, advanced & trusted computed, scalable computing & communications, cloud & big data computing, Internet of people and smart city innovation (smartworld/SCAL-COM/UIC/ATC/CBDcom/IOP/SCI)*, pp. 1–6. IEEE, 2017.
- Chaofan Chen, Oscar Li, Daniel Tao, Alina Barnett, Cynthia Rudin, and Jonathan K Su. This looks like that: deep learning for interpretable image recognition. *Advances in neural information processing systems*, 32, 2019.
- Ricky TQ Chen, Yulia Rubanova, Jesse Bettencourt, and David K Duvenaud. Neural ordinary differential equations. *Advances in neural information processing systems*, 31, 2018.
- Yu Cheng, Fei Wang, Ping Zhang, and Jianying Hu. Risk prediction with electronic health records: A deep learning approach. In *Proceedings of the 2016 SIAM international conference on data mining*, pp. 432–440. SIAM, 2016.
- Edward Choi, Mohammad Taha Bahadori, Jimeng Sun, Joshua Kulas, Andy Schuetz, and Walter Stewart. Retain: An interpretable predictive model for healthcare using reverse time attention mechanism. *Advances in neural information processing systems*, 29, 2016.
- Li Deng and Yang Liu. *Deep learning in natural language processing*. Springer, 2018.
- Daehoon Gwak, Gyuhyeon Sim, Michael Poli, Stefano Massaroli, Jaegul Choo, and Edward Choi. Neural ordinary differential equations for intervention modeling. *arXiv preprint arXiv:2010.08304*, 2020.
- JB Heaton, Nicholas G Polson, and Jan Hendrik Witte. Deep learning in finance. *arXiv preprint arXiv:1602.06561*, 2016.
- David A Jenkins, Matthew Sperrin, Glen P Martin, and Niels Peek. Dynamic models to predict health outcomes: current status and methodological challenges. *Diagnostic and prognostic research*, 2: 1–9, 2018.
- Ming Jin, Yu Zheng, Yuan-Fang Li, Siheng Chen, Bin Yang, and Shirui Pan. Multivariate time series forecasting with dynamic graph neural odes. *IEEE Transactions on Knowledge and Data Engineering*, 2022.
- Alistair Johnson and et al. MIMIC-IV-ED (version 2.2). PhysioNet, 2023. URL <https://doi.org/10.13026/5ntk-km72>.
- Patrick Kidger. On neural differential equations. *arXiv preprint arXiv:2202.02435*, 2022.
- Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural controlled differential equations for irregular time series. *Advances in Neural Information Processing Systems*, 33:6696–6707, 2020.
- Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521(7553):436–444, 2015.

- Simon A. Lee and Timothy Lindsey. Can large language models abstract medical coded language? *arXiv preprint arXiv:2403.10822*, 2024a.
- Simon A Lee and Timothy Lindsey. Do large language models understand medical codes? *arXiv preprint arXiv:2403.10822*, 2024b.
- Simon A. Lee, Trevor Brokowski, and Jeffrey N. Chiang. Enhancing antibiotic stewardship using a natural language approach for better feature representation. *arXiv preprint arXiv:2405.20419*, 2024a.
- Simon A Lee, Sujay Jain, Alex Chen, Arabdha Biswas, Jennifer Fang, Akos Rudas, and Jeffrey N Chiang. Multimodal clinical pseudo-notes for emergency department prediction tasks using multiple embedding model for ehr (meme). *arXiv preprint arXiv:2402.00160*, 2024b.
- Xuhong Li, Haoyi Xiong, Xingjian Li, Xuanyu Wu, Xiao Zhang, Ji Liu, Jiang Bian, and Dejing Dou. Interpretable deep learning: Interpretation, interpretability, trustworthiness, and beyond. *Knowledge and Information Systems*, 64(12):3197–3234, 2022.
- Alec J Linot, Joshua W Burby, Qi Tang, Prasanna Balaprakash, Michael D Graham, and Romit Maulik. Stabilized neural ordinary differential equations for long-time forecasting of dynamical systems. *Journal of Computational Physics*, 474:111838, 2023.
- Jingshu Liu, Zachariah Zhang, and Narges Razavian. Deep ehr: Chronic disease prediction using medical notes. In *Machine Learning for Healthcare Conference*, pp. 440–464. PMLR, 2018.
- Riccardo Miotto, Fei Wang, Shuang Wang, Xiaoqian Jiang, and Joel T Dudley. Deep learning for healthcare: review, opportunities and challenges. *Briefings in bioinformatics*, 19(6):1236–1246, 2018.
- Kyoka Ono and Simon A. Lee. Text serialization and their relationship with the conventional paradigms of tabular machine learning. *arXiv preprint arXiv:2406.13846*, 2024.
- Zhaozhi Qian, William Zame, Lucas Fleuren, Paul Elbers, and Mihaela van der Schaar. Integrating expert odes into neural odes: pharmacology and disease progression. *Advances in Neural Information Processing Systems*, 34:11364–11383, 2021.
- Muhammad Imran Razzak, Saeeda Naz, and Ahmad Zaib. Deep learning for medical image processing: Overview, challenges and the future. *Classification in BioApps: Automation of decision making*, pp. 323–350, 2018.
- Domenec Ruiz-Balet and Enrique Zuazua. Neural ode control for classification, approximation, and transport. *SIAM Review*, 65(3):735–773, 2023.
- Gregor Stiglic, Primoz Kocbek, Nino Fijacko, Marinka Zitnik, Katrien Verbert, and Leona Cilar. Interpretability of machine learning-based prediction models in healthcare. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, 10(5):e1379, 2020.
- Anant Subramanian, Danish Pruthi, Harsh Jhamtani, Taylor Berg-Kirkpatrick, and Eduard Hovy. Spine: Sparse interpretable neural embeddings. In *Proceedings of the AAAI conference on artificial intelligence*, volume 32, 2018.
- Neil C Thompson, Kristjan Greenewald, Keeheon Lee, and Gabriel F Manso. The computational limits of deep learning. *arXiv preprint arXiv:2007.05558*, 2020.
- Athanassios Voulodimos, Nikolaos Doulamis, Anastasios Doulamis, and Eftychios Protopapadakis. Deep learning for computer vision: A brief review. *Computational intelligence and neuroscience*, 2018, 2018.
- Eric Wu, Kevin Wu, Roxana Daneshjou, David Ouyang, Daniel E Ho, and James Zou. How medical ai devices are evaluated: limitations and recommendations from an analysis of fda approvals. *Nature Medicine*, 27(4):582–584, 2021.
- Jionglin Wu, Jason Roy, and Walter F Stewart. Prediction modeling using ehr data: challenges, strategies, and a comparison of machine learning approaches. *Medical care*, 48(6):S106–S113, 2010.

Quan-shi Zhang and Song-Chun Zhu. Visual interpretability for deep learning: a survey. *Frontiers of Information Technology & Electronic Engineering*, 19(1):27–39, 2018.

Yu Zhang, Peter Tiňo, Aleš Leonardis, and Ke Tang. A survey on neural network interpretability. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 5(5):726–742, 2021.

Juntang Zhuang, Nicha Dvornek, Xiaoxiao Li, Sekhar Tatikonda, Xenophon Papademetris, and James Duncan. Adaptive checkpoint adjoint method for gradient estimation in neural ode. In *International Conference on Machine Learning*, pp. 11639–11649. PMLR, 2020.

A Appendix

A.1 Neural ODEs Architecture

Fundamentally a Neural ODE is easy to implement. The first block defines `ODEFunc`, a PyTorch module representing the derivative function of a Neural ODE, which applies a linear transformation followed by a ReLU activation to its input. The second block introduces `ODEBlock`, another PyTorch module that integrates the derivative function defined in `ODEFunc` over time, from $t = 0$ to $t = 1$, using the `odeint` solver to produce the system's state at $t = 1$. The complexity of the model can be further extended for potential better performance, but the simplicity in this model also showed strong performance as seen in the study.

```
# Define the ODE function (derivative)
class ODEFunc(nn.Module):
    def __init__(self, dim):
        super().__init__()
        self.linear = nn.Linear(dim, dim)
        self.relu = nn.ReLU(inplace=True)

    def forward(self, t, x):
        return self.relu(self.linear(x))
```

```
# Define the ODE block that integrates
# the ODEFunc
class ODEBlock(nn.Module):
    def __init__(self, odefunc):
        super().__init__()
        self.odefunc = odefunc

    def forward(self, x):
        # Integrate from t=0 to t=1 \
        return odeint(self.odefunc, x, \
                      torch.tensor([0, 1]), \
                      dtype=torch.float32), \
                      method='dopri5')[1]
```

A.2 Understanding the Methodology through MNIST

Neural Ordinary Differential Equations for MNIST Image Classification In this section, we explore the application of neural ordinary differential equations (neural ODEs) for the MNIST image classification task. The MNIST dataset consists of handwritten digits from 0 to 9, and the goal is to build a model that can accurately classify the given images into their respective digit classes.

Let $\mathcal{X} = \mathbb{R}^{28 \times 28}$ be the input space of MNIST images and $\mathcal{Y} = \{0, 1, \dots, 9\}$ be the set of digit class labels. Our objective is to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ that can predict the digit class of a given MNIST image.

We can formulate the MNIST image classification task as a neural ODE problem, where the dynamics of the image representation are modeled by an ODE:

$$\frac{dx(t)}{dt} = f(x(t), t, \theta)$$

where $x(t) \in \mathbb{R}^d$ is the hidden state representing the image at time t , $f : \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^p \rightarrow \mathbb{R}^d$ is the neural network-based function that governs the dynamics of the image representation, and $\theta \in \mathbb{R}^p$ are the parameters of the neural network.

To train the neural ODE model, we can use a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathcal{X}$ is the input MNIST image and $y_i \in \mathcal{Y}$ is the corresponding digit class label. We can define the following loss function:

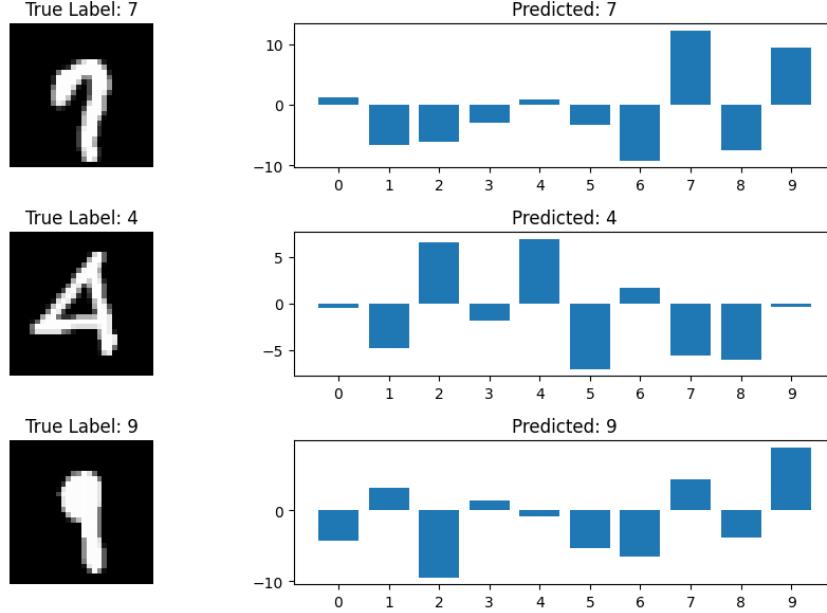


Figure 2: Showing the Probabilities of the Input Image on the MNIST Dataset during inference

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^N \ell(f(x_i, \theta), y_i)$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is the cross-entropy loss function.

To optimize the parameters θ , we can use gradient-based methods, such as gradient descent or its variants. The gradients of the loss function with respect to the parameters θ can be computed using the adjoint method, which has a time complexity of $\mathcal{O}(k \cdot d)$, where k is the number of stages in the numerical method used to solve the ODE.

Once the neural ODE model is trained, we can use it to predict the digit class of a new MNIST image x by solving the ODE:

$$\frac{dx(t)}{dt} = f(x(t), t, \theta^*)$$

where θ^* are the learned parameters. The final digit class prediction can be obtained by applying a softmax layer to the terminal state $x(T)$, where T is the final time point.

The key advantage of using a neural ODE for MNIST image classification is its ability to capture the dynamic evolution of the image representation, which can be particularly useful for modeling the spatial and temporal dependencies in the image data. Moreover, the continuous-time nature of the ODE allows for more flexible and expressive representations compared to traditional discrete-time models.

A.3 Neural ODEs: Overcoming the Exploding Gradients Dilemma in Text Classification

One of the key advantages of neural ordinary differential equations for text classification is their ability to mitigate the problem of exploding gradients, a common issue that plagues many traditional deep learning architectures. Exploding gradients occur when the gradients of the loss function with respect to the model parameters grow exponentially during the training process, leading to numerical instability and poor model performance.

The inherent structure of neural ODEs can be mathematically formulated as follows. Consider a neural ODE of the form:

$$\frac{dx(t)}{dt} = f(x(t), t, \theta)$$

where $x(t)$ is the state of the system at time t , f is the neural network-based function that governs the dynamics of the system, and θ is the set of parameters of the neural network.

The key property that allows neural ODEs to overcome the exploding gradients problem is the smoothness and stability of the ODE formulation. Unlike traditional recurrent neural networks (RNNs) or deep feedforward networks, which can suffer from vanishing or exploding gradients due to the compounding of operations across many layers, neural ODEs leverage the stability and convergence properties of ordinary differential equations.

Mathematically, the gradients of the neural ODE solution $x(t)$ with respect to the parameters θ can be computed using the adjoint method, which involves solving an additional ODE system:

$$\frac{da(t)}{dt} = -a(t)^\top \frac{\partial f}{\partial x}(x(t), t, \theta)$$

where $a(t)$ is the adjoint state.

The key difference is that the gradients in the neural ODE formulation do not grow exponentially, but rather evolve smoothly over time, preserving the stability of the training process. This is due to the fact that the gradients are governed by the continuous-time dynamics of the ODE system, rather than being compounded across discrete layers.

A.4 Convergence Proof for Training Neural Ordinary Differential Equations

When training neural networks to model dynamical systems it is crucial to establish theoretical guarantees on the convergence of the training process. To achieve this, we provide a rigorous convergence proof for training neural ODEs.

The neural ODE is defined as $\frac{dx(t)}{dt} = f(x(t), t, \theta)$, where $x(t)$ is the state of the system, f is the neural network-based function that governs the dynamics, and θ is the set of parameters. The training problem is formulated as an optimization problem, where the goal is to minimize the loss function $L(\theta) = \frac{1}{N} \sum_{i=1}^N \|x(t_i) - x_i\|^2$, where x_i are the observed states at times t_i .

To prove the convergence, we make the following assumptions: (1) the neural network function f is Lipschitz continuous with respect to x and θ , (2) the neural ODE has a unique solution $x(t)$ for any given initial condition and parameters, and (3) the optimization algorithm used to train the neural network converges to a stationary point of the loss function. Under these assumptions, we show that the neural ODE solution $x(t)$ will converge to the true state of the system as the number of training iterations goes to infinity.

A.5 Time Complexity

The time complexity of training a neural ODE is analyzed in terms of the forward pass (solving the ODE to obtain $x(t_i)$), the backward pass (computing the gradients), and the optimization algorithm. The overall time complexity is $\mathcal{O}(N \cdot k \cdot n / \varepsilon)$, where N is the number of data points, k is the number of stages in the numerical method used to solve the ODE, n is the number of state variables, and ε is the desired accuracy of the optimization algorithm. The key factors that influence the time complexity are the number of data points, the complexity of the ODE solver, and the convergence rate of the optimization algorithm.

A.6 Call for more sophisticated Neural ODE models

In this work, we showcased a simple demonstration of how these models are interpretable and the advantages to modeling text continuously. Our NODE model, while not very sophisticated,

demonstrates the potential of this approach. We see many directions for improvement, such as trying different ODE solvers, that could further optimize this model and make it an even better predictor.

We therefore encourage researchers to build upon this foundational NODE model and explore various avenues to enhance its performance. Some potential directions include experimenting with alternative ODE solvers, which may lead to improved stability, accuracy, and computational efficiency. Another potential direction is leveraging auxiliary objectives, such as language modeling or semantic understanding, can help the NODE model learn more robust and transferable representations.

The versatility and dynamic modeling capabilities of neural ODEs make them a promising avenue for pushing the direction of interpretable text classification and unlocking new research in natural language processing.

A.7 Sentiment Analysis with Neural Ordinary Differential Equations

Sentiment analysis is a fundamental task in natural language processing, where the goal is to classify a given text as expressing a positive, negative, or neutral sentiment. In this section of the appendix, we present a theoretical framework for sentiment analysis using neural ordinary differential equations (neural ODEs) to help bridge the gap between text classification and NODEs.

Let \mathcal{X} be the input space of text documents and $\mathcal{Y} = \{-1, 0, 1\}$ be the set of sentiment labels, where -1 represents negative sentiment, 0 represents neutral sentiment, and 1 represents positive sentiment. Our objective is to learn a function $f : \mathcal{X} \rightarrow \mathcal{Y}$ that can accurately predict the sentiment of a given text.

We can formulate the sentiment analysis task as a neural ODE problem, where the dynamics of the text representation are modeled by an ODE:

$$\frac{dx(t)}{dt} = f(x(t), t, \theta)$$

where $x(t) \in \mathbb{R}^d$ is the hidden state representing the text at time t , $f : \mathbb{R}^d \times \mathbb{R} \times \mathbb{R}^p \rightarrow \mathbb{R}^d$ is the neural network-based function that governs the dynamics of the text representation, and $\theta \in \mathbb{R}^p$ are the parameters of the neural network.

To train the neural ODE model, we can use a dataset $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^N$, where $x_i \in \mathcal{X}$ is the input text and $y_i \in \mathcal{Y}$ is the corresponding sentiment label. We can define the following loss function:

$$\mathcal{L}(\theta) = \frac{1}{N} \sum_{i=1}^N \ell(f(x_i, \theta), y_i)$$

where $\ell : \mathcal{Y} \times \mathcal{Y} \rightarrow \mathbb{R}$ is a suitable loss function, such as the cross-entropy loss.

To optimize the parameters θ , we can use gradient-based methods, such as gradient descent or its variants. The gradients of the loss function with respect to the parameters θ can be computed using the adjoint method, which has a time complexity of $\mathcal{O}(k \cdot d)$, where k is the number of stages in the numerical method used to solve the ODE.

Once the neural ODE model is trained, we can use it to predict the sentiment of a new text x by solving the ODE:

$$\frac{dx(t)}{dt} = f(x(t), t, \theta^*)$$

where θ^* are the learned parameters. The final sentiment prediction can be obtained by applying a classifier to the terminal state $x(T)$, where T is the final time point.

The key advantage of using a neural ODE for sentiment analysis is its ability to capture the dynamic evolution of the text representation, which can be particularly useful for modeling the temporal and contextual aspects of language. Moreover, the continuous-time nature of the ODE allows for more flexible and expressive representations compared to traditional discrete-time models.

Model	Interpretable?	Accuracy	F1	AUROC
Logistic Regression	yes	0.655	0.654	0.702
LGBM	yes	0.656	0.655	0.818
LSTM	no	0.671	0.672	0.753
BERT	no	0.737	0.738	0.799
Neural ODE	yes	0.731	0.721	0.7209

Table 4: Benchmark of text classification (Discharge Location). Prevalence of this label was approximately 45%.

Model	Interpretable?	Accuracy	F1	AUROC
Logistic Regression	yes	0.971	0.095	0.768
LGBM	yes	0.973	0.127	0.815
LSTM	no	0.965	0.067	0.512
BERT	no	0.988	0.072	0.546
Neural ODE	yes	0.970	0.069	0.529

Table 5: Benchmark of text classification (Predicting Mortality). Prevalence of this label was approximately 3%.

Model	Interpretable?	Accuracy	F1	AUROC
Logistic Regression	yes	0.835	0.427	0.807
LGBM	yes	0.877	0.545	0.818
LSTM	no	0.881	0.543	0.799
BERT	no	0.898	0.572	0.870
Neural ODE	yes	0.880	0.552	0.728

Table 6: Benchmark of text classification. (ICU Requirement) Prevalence of this label was approximately 19%.